(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 4 455 921 A1**

(12) **EUROPEAN PATENT APPLICATION**
published in accordance with Art. 153(4) EPC

(43) Date of publication:
30.10.2024 Bulletin 2024/44

(21) Application number: **23856087.4**

(22) Date of filing: **17.04.2023**

(51) International Patent Classification (IPC):
G06F 21/60 *(2013.01)*

(52) Cooperative Patent Classification (CPC):
G06F 21/53; G06F 21/60; H04L 9/40; H04L 51/046; H04L 67/55; H04L 67/566

(86) International application number:
PCT/CN2023/088670

(87) International publication number:
WO 2024/040992 (29.02.2024 Gazette 2024/09)

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC ME MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA
Designated Validation States:
KH MA MD TN

(30) Priority: **22.08.2022 CN 202211008092**
07.02.2023 CN 202310129070

(71) Applicant: **Honor Device Co., Ltd.**
Shenzhen, Guangdong 518040 (CN)

(72) Inventor: **LI, Ping**
Shenzhen, Guangdong 518040 (CN)

(74) Representative: **Goddar, Heinz J.**
Boehmert & Boehmert
Anwaltspartnerschaft mbB
Pettenkoferstrasse 22
80336 München (DE)

(54) MESSAGE PROCESSING METHOD AND APPARATUS AND ELECTRONIC DEVICE

(57) Embodiments of this application provide a message processing method and apparatus, and an electronic device. The method is applied to a terminal device, and the method includes: receiving a first message; and pulling up a first application when the first message is a reassembly message, so that the first application performs message reassembly on the first message, including: reading, by the first application, message reassembly configuration information, and performing a message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message. According to the method in the embodiments of this application, after a pushed message is received, the first application is pulled up to reassemble the message, so as to improve user experience of message display.

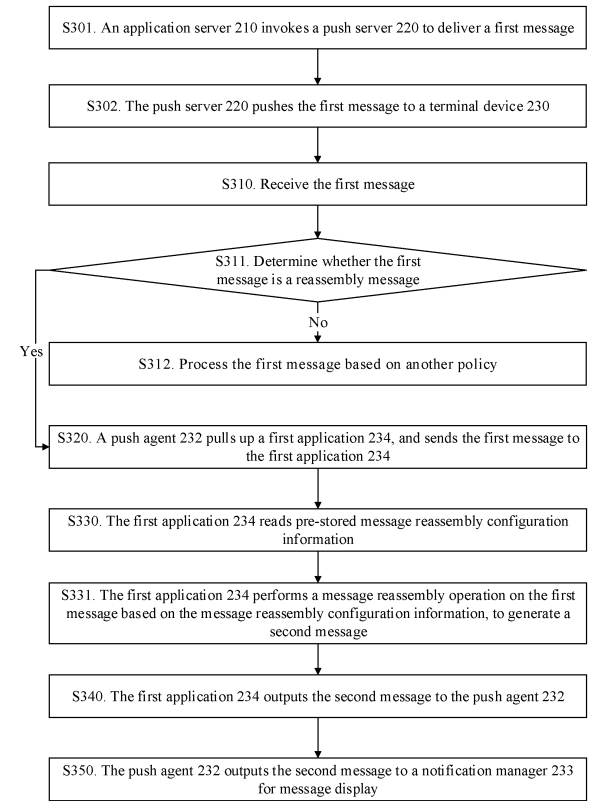

FIG. 3

EP 4 455 921 A1

Description

TECHNICAL FIELD

[0001] This application relates to the field of computer technologies, and in particular, to a message processing method and apparatus, and an electronic device.

BACKGROUND

[0002] Nowadays, terminal devices including a mobile phone, a tablet computer, a watch, a television, a personal computer, and the like provide a push (Push) capability of sending a message from a cloud server to a terminal. The implementation of a Push message generally includes two types: a notification message and a transparent message. For the transparent message, an application is directly pulled up, and data is transferred to the application. The application can make different behaviors based on the transferred data. For the notification message, an application is not pulled up, and a system component displays the notification message based on data content of Push, for example, displays the notification message in a system notification area (for example, prompts a user with push notification information of the application in a pull-down notification bar of Android/iOS or a tile of Windows).

[0003] For the notification message, the application is not pulled up. The notification message is designed in such a way to control resource occupation of the terminal device and reduce interference to the user. However, in the design solution, content transferred in the Push notification message needs to be plaintext and cannot be modified. As a result, the terminal device cannot protect sensitive information in the notification message, and cannot provide personalized display of the notification message.

SUMMARY

[0004] In view of a problem of a push notification message in the conventional technology, this application provides a message processing method and apparatus, and an electronic device, and this application further provides a computer-readable storage medium.

[0005] The following technical solutions are used in embodiments of this application:

[0006] According to a first aspect, this application provides a message processing method, where the method is applied to a terminal device, and the method includes:

receiving a first message; and

pulling up a first application when the first message is a reassembly message, so that the first application performs message reassembly on the first message, including:

reading, by the first application, message reassembly configuration information, where the message reassembly configuration information is information configured by a second application for the first application, an operation capable of being implemented by the second application includes at least a message reassembly operation, and the message reassembly configuration information is used to support the first application in implementing the message reassembly operation; and

performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message.

[0007] According to the method in the embodiments of this application, after a pushed message is received, the first application is pulled up to reassemble the message, so as to improve user experience of message display. Further, the first application is only used to implement the message reassembly operation of the second application, and the first application does not need to have a complete structure of the second application. Therefore, a hardware resource occupied by pulling up the first application is fewer than a hardware resource occupied by pulling up the second application. This effectively controls hardware resource occupation of the terminal device while improving user experience of notification message display.

[0008] In an implementation of the first aspect, to improve data security and avoid leakage of the first message, the first application is a trusted application.

[0009] In an implementation of the first aspect, to improve data security and avoid leakage of the message reassembly configuration information, the message reassembly configuration information is stored in a trusted credential of the first application.

[0010] Because the message reassembly configuration information is stored in the trusted credential of the first application, an application service that pulls up the first application cannot access the trusted credential of the first application. Therefore, the application service that pulls up the first application cannot obtain the message reassembly configuration information, thereby ensuring security of performing the message reassembly operation.

[0011] In an implementation of the first aspect, the message reassembly configuration information includes message reassembly code and an initialization parameter; and

the performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message includes:

executing, by the first application, the message reassembly code for the first message based on the initialization parameter, to generate the second mes-

sage.

[0012] In an implementation of the first aspect, before the receiving a first message, the method further includes:
invoking, by the second application, the first application, to set the message reassembly configuration information.

[0013] In an implementation of the first aspect, to improve security, before the receiving a first message, the method further includes: obtaining, from a cloud, a range of resources and interfaces that the first application is allowed to access; and
the performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message includes:
performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information under a limitation of the range of resources and interfaces that the first application is allowed to access, to generate the second message.

[0014] Based on the limitation of the range of accessed resources and interfaces, in a process in which the message reassembly operation is performed in the first application, only a controllable interface such as a basic data operation, encryption/decryption, and a corresponding application resource query is provided, and the first application cannot pull up another application and access another controlled resource, thereby improving system running security.

[0015] Specifically, in an implementation of the first aspect, the obtaining, from a cloud, a range of resources and interfaces that the first application is allowed to access includes: obtaining, by a push agent from the cloud, the range of resources and interfaces that the first application is allowed to access.

[0016] In an implementation of the first aspect, the method further includes:
outputting the second message for display.

[0017] In an implementation of the first aspect, the outputting the second message for display includes:

outputting, by the first application, the second message to a push agent; and
outputting, by the push agent, the second message to a notification manager for message display.

[0018] In an implementation of the first aspect, the method further includes:
if an exception occurs when the first application performs the message reassembly operation, preventing display of the second message.

[0019] In an implementation of the first aspect,

the receiving a first message includes: receiving, by a push agent, the first message; and
the invoking, when the first message is a reassembly message, a first application to perform message reassembly on the first message includes: when the push agent determines that the first message is a reassembly message, pulling up, by the push agent, the first application, so that the first application performs message reassembly on the first message.

[0020] In an implementation of the first aspect, before the receiving a first message, the method further includes:

applying, by the second application, to the push agent for a push token;
after receiving the application of the second application, setting, by the push agent, to allow the second application to invoke the first application, and returning, by the push agent, the push token to the second application;
after the second application obtains the push token, invoking, by the second application, the first application, to set the message reassembly configuration information for the first application; and
storing, by the first application, the message reassembly configuration information.

[0021] In an implementation of the first aspect, to improve data security, the first message is end-to-end encrypted information; and
in a process in which the first application performs the message reassembly operation on the first message, the first application invokes an encryption/decryption interface to decrypt the first message.

[0022] The application service that pulls up the first application does not participate in a decryption operation, and therefore cannot obtain specific content of the first message. In an entire process from receiving the pushed first message to displaying the second message, the application service that pulls up the first application can obtain only content obtained after message reassembly, thereby greatly improving message security.

[0023] According to a second aspect, this application further provides a message processing apparatus, where the apparatus is applied to a terminal device, and the apparatus includes:

a receiving module, configured to receive a first message;
a message reassembly module, configured to pull up a first application module when the first message is a reassembly message; and
the first application module, configured to:

read message reassembly configuration information, where the message reassembly configuration information is information configured by a second application module for the first application module, an operation capable of being

implemented by the second application module includes at least a message reassembly operation, and the message reassembly configuration information is used to support the first application module in implementing the message reassembly operation; and

perform the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message.

[0024] According to a third aspect, this application further provides an electronic device, where the electronic device includes a memory configured to store a computer program instruction and a processor configured to execute a computer program instruction, and when the computer program instruction is executed by the processor, the electronic device is triggered to perform the steps of the method according to the first aspect.

[0025] According to a fourth aspect, this application further provides a computer-readable storage medium, where the computer-readable storage medium stores a computer program, and when the computer program is run on a computer, the computer is enabled to perform the method according to the first aspect.

BRIEF DESCRIPTION OF DRAWINGS

[0026]

FIG. 1 is a schematic diagram of a message push procedure according to an embodiment of this application;

FIG. 2 is a schematic diagram of a message push application scenario according to an embodiment of this application;

FIG. 3 is a flowchart of a message processing method according to an embodiment of this application;

FIG. 4 is a flowchart of a message processing method according to an embodiment of this application;

FIG. 5 is a schematic diagram of a message processing apparatus according to an embodiment of this application; and

FIG. 6 is a schematic diagram of an electronic device according to an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

[0027] To make the objective, technical solutions, and advantages of this application clearer, the technical solutions of this application are clearly and completely described below with reference to specific embodiments of this application and corresponding accompanying drawings. It is clear that the described embodiments are merely some but not all of embodiments of this application. Based on the embodiments of this application, all other embodiments obtained by a person of ordinary skill in the art without creative efforts fall within the protection scope of this application.

[0028] Terms used in the DESCRIPTION OF EMBODIMENTS section of this application are only used to explain specific embodiments of this application, and are not intended to limit this application.

[0029] FIG. 1 is a schematic diagram of a message push procedure according to an embodiment of this application.

[0030] As shown in FIG. 1, when a service server 111 and a big data operation platform 112 need to push a message to a terminal device 120, the service server 111 and the big data operation platform 112 first push the message to a push server (PushServer) 113.

[0031] The push server 113 sends, to the terminal device 120, the message that needs to be pushed.

[0032] A push core (Push Core) 121 of the terminal device 120 receives the message pushed by the Push server 113, and classifies the received push message.

[0033] If the push message is a transparent message, an application (App) 122 that needs to receive the transparent message is pulled up. The application 122 receives the transparent message based on a push function module 123 (push software development kit (Push Software Development Kit, Push SDK)) of the application 122. Then, the application 122 performs corresponding information processing on the transparent message.

[0034] When the push message is a transparent message, the terminal device 120 needs to pull up the application 122 used to process the transparent information. Starting and running of the application 122 necessarily occupy a hardware resource of the terminal device 120.

[0035] If the push message is a notification message, the push core 121 directly sends the notification message to a push agent (Push Agent) 124, and the push agent 124 sends the notification message to a notification manager (NotificationManager) 125 for message display.

[0036] When the push message is a notification message, the terminal device 120 does not need to pull up an application used to process the message, and an increase in hardware resource occupation is avoided. However, because the push agent 124 directly sends the notification message to the notification manager 125 for message display, the terminal device 120 can display only content of the pushed notification message fully, and cannot perform personalized content modification. As a result, the notification message cannot be displayed in a personalized manner, and sensitive information is shielded.

[0037] Further, the push agent 124 directly sends the notification message to the notification manager 125 for message display, and the terminal device 120 does not pull up the application to process the notification message. Therefore, the terminal device 120 cannot perform a decryption operation on the notification message, and the notification message needs to be plaintext in an entire transmission process. In a process of transmitting the notification message, the notification message is easily leaked.

[0038] For example, in a data transmission process (101) in which the service server 111 pushes the notification message to the Push server 113, in a data transmission process (102) in which the big data operation platform 112 pushes the notification message to the Push server 113, in a data transmission process (103) in which the Push server 113 pushes the notification message to the push core 121 of the terminal device 120, and in a data transmission process (104) in which the push core 121 sends the notification message to the push agent 124, if a data transmission protocol is cracked, a cracker may obtain content of the plaintext notification message.

[0039] For another example, the service server 111 and the big data operation platform 112 push the plaintext notification message to the Push server 113, and the Push server 113 pushes the plaintext notification message to the push core 121 of the terminal device 120. Therefore, the notification message temporarily stored in the Push server 113 is also plaintext. If the Push server 113 is attacked, content of the plaintext notification message may be leaked.

[0040] In view of the foregoing case, an embodiment of this application provides a message processing method, and the method is applied to a terminal device. In the message processing method in an embodiment of this application, after a pushed notification message is received, an application is pulled up to reassemble the notification message (for example, delete sensitive information, and reassemble a text description manner of the message based on a personalized requirement of a user), and the reassembled notification message is output and displayed, thereby improving user experience of notification message display.

[0041] Further, when the pushed message is a transparent message, the terminal device pulls up an application (a second application) for the transparent message, to process the transparent message. Therefore, based on an implementation solution of the transparent message, when the second application has a function of implementing a message reassembly operation, and the terminal device receives a notification message that needs to be reassembled, the terminal device may pull up the second application, so that the second application reassembles the notification message, and then the second application outputs and displays the reassembled message. Although the foregoing solution may improve user experience of notification message display, because the second application needs to be pulled up, hardware resource occupation of the terminal device is increased.

[0042] In view of the foregoing problem that hardware resource occupation of the terminal device is increased, in the message processing method in an embodiment of this application, a first application is constructed based on the second application, and the first application is used to implement the message reassembly operation of the second application. In this way, when the terminal device receives the notification message that needs to be reassembled, the terminal device may pull up the first application, so that the first application reassembles the notification message, and then the first application outputs and displays the reassembled message. The first application is only used to implement the message reassembly operation of the second application, and the first application does not need to have a complete structure of the second application. Therefore, a hardware resource occupied by pulling up the first application is fewer than a hardware resource occupied by pulling up the second application. This effectively controls hardware resource occupation of the terminal device while improving user experience of notification message display.

[0043] FIG. 2 is a schematic diagram of a message push application scenario according to an embodiment of this application. FIG. 3 is a flowchart of a message processing method according to an embodiment of this application.

[0044] As shown in FIG. 2 and FIG. 3, in the message push application scenario, an application (App) server 210 of a cloud invokes a push server (PushServer) 220 to deliver a first message (for example, a transparent message or a notification message) (S301). The push server 220 pushes, to a terminal device 230, the first message delivered by the application server 210 (S302).

[0045] The terminal device 230 performs the following procedure shown in FIG. 2, to implement message processing.

[0046] S310. Apush core 231 of the terminal device 230 receives the first message pushed by the push server 220.

[0047] S311. The terminal device 230 determines whether the first message is a reassembly message.

[0048] If the first message is not a reassembly message, the first message is processed based on another policy (S312).

[0049] If the first message is a reassembly message, a first application 234 is pulled up, so that the first application 234 performs a message reassembly operation on the first message (S320).

[0050] Specifically, the message reassembly operation may include a plurality of pieces of different operation content. This is not limited in this application. For example, a combination manner of elements in the message is rearranged, a message style is changed, a message icon and background are added, and a specified element/variable in the message is replaced. For example, the delivered message is: "Dear {name}, the {service} you purchased is about to expire...", and variables {name} and {service} in the message are replaced in a message reassembly process.

[0051] Specifically, in an actual application scenario, the first information may be marked as a reassembly message in a plurality of manners.

[0052] For example, in an implementation, the reassembly message is marked by adding a mark. In a process in which the application server 210 delivers the first message, if the first message is a reassembly message,

a first mark used to mark the reassembly message is added into the first information.

[0053] For another example, in another implementation, the reassembly message is marked by binding an information source. That the application server 210 is an information source of the reassembly message is recorded at the terminal device 230. The first message delivered by the application server 210 includes information source information. If the terminal device 230 identifies that an information source of the first message is the application server 210, the terminal device 230 may determine that the first message is a reassembly message.

[0054] In an implementation of S311, the reassembly message is a notification message on which the message reassembly operation needs to be performed.

[0055] In S311, the push core 231 of the terminal device 230 determines a type of the first message (a notification message or a transparent message). If the first message is a transparent message (that is, it may be determined that the first message is not a reassembly message), the push core 231 pulls up a corresponding application based on a processing policy of the transparent message, to process the first message. If the first message is a notification message, the push core 231 sends the first message to a push agent 232. The push agent 232 determines whether the message reassembly operation needs to be performed on the first message (for example, performs determining by determining whether the first mark is included, or performs determining by using the information source). If the message reassembly operation does not need to be performed on the first message (that is, it may be determined that the first message is not a reassembly message), the push agent 232 sends the first message to a notification manager 233 for message display.

[0056] If the message reassembly operation needs to be performed on the first message (the first message is a reassembly message), the push agent 232 performs S320.

[0057] In another implementation of S311, the reassembly message is a notification message on which the message reassembly operation needs to be performed or a transparent message on which only the message reassembly operation needs to be performed.

[0058] In S311, the push core 231 of the terminal device 230 determines whether the message reassembly operation needs to be performed on the first message (for example, performs determining by determining whether the first mark is included, or performs determining by using the information source).

[0059] If the message reassembly operation needs to be performed on the first message, the push core 231 adds a second mark into the first message, and sends, to a push agent 232, the first message into which the second mark is added. After the push agent 232 identifies that the first message has the first mark, the push agent 232 performs S320.

[0060] If the message reassembly operation does not need to be performed on the first message (that is, it may be determined that the first message is not a reassembly message), the push core 231 determines a type of the first message (a notification message or a transparent message). If the first message is a transparent message, the push core 231 pulls up a corresponding application based on a processing policy of the transparent message, to process the first message. If the first message is a notification message, the push core 231 sends the first message to a push agent 232. After identifying that the first message does not have the first mark, the push agent 232 sends the first message to a notification manager 233 for message display.

[0061] S320. The push agent 232 pulls up the first application 234, and sends the first message to the first application 234.

[0062] Specifically, to improve data security and avoid leakage of the first message, in an embodiment, the first application 234 is a trusted application (Trusted Application, TA), and runs in a trusted execution environment (Trusted execution environment, TEE) in a terminal system operating system. The push agent 232 is a client applicant (Client Applicant, CA) that can invoke a trusted application (first application 234).

[0063] S330. The first application 234 reads pre-stored message reassembly configuration information.

[0064] Specifically, to improve data security and avoid leakage of the message reassembly configuration information, in an embodiment, the message reassembly configuration information is stored in a trusted credential 235 of the first application 234 (TA).

[0065] Because the message reassembly configuration information is stored in the trusted credential 235 of the first application 234, the push agent 232 can pull up only the first application 234, but cannot access the trusted credential 235 of the first application 234. Therefore, the push agent 232 cannot obtain the message reassembly configuration information, thereby ensuring security of performing the message reassembly operation.

[0066] Further, in another embodiment, the message reassembly configuration information may be stored at another location.

[0067] S331. The first application 234 performs the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message.

[0068] After S331, the first application 234 outputs the second message for display.

[0069] S340. The first application 234 outputs the second message to the push agent 232.

[0070] S350. The push agent 232 outputs the second message to the notification manager 233 for message display.

[0071] Specifically, in S330 and S331, for different message reassembly configuration information, the first application 234 may implement different message reassembly operations. For example, based on first message reassembly configuration information, the first applica-

tion 234 is run to perform the message reassembly operation on the message, so as to delete a sensitive word in the message. Based on second message reassembly configuration information, the first application is run to perform the message reassembly operation on the message, so as to split a long-sentence message into a short-sentence message.

[0072] Specifically, in an embodiment, the message reassembly configuration information includes message reassembly code and an initialization parameter. In S331, the first application 234 executes the read message reassembly code for the first message based on the read initialization parameter, to generate the second message.

[0073] Further, in another embodiment, the message reassembly configuration information may be of another different content structure.

[0074] For example, in another embodiment, a unified initialization parameter is built in the first application 234, and the message reassembly configuration information includes message reassembly code. In S331, the first application 234 executes the read message reassembly code for the first message based on the built-in initialization parameter, to generate the second message.

[0075] For another example, in another embodiment, the first application 234 includes message reassembly code, and the message reassembly configuration information includes an initialization parameter. In S331, the first application 234 executes the read message reassembly code in the first application for the first message based on the read initialization parameter, to generate the second message.

[0076] Further, in S331, in addition to performing the message reassembly operation on the first message, the first application 234 may further perform another processing operation.

[0077] For example, in an embodiment, to reduce a data transmission amount, in S301, the message delivered by the application server 210 to the push server 220 is compressed data. Alternatively, in S301, the message delivered by the application server 210 to the push server 220 is uncompressed data. After receiving the message, the push server 220 compresses the message, and then pushes the compressed message to the terminal device 230.

[0078] In S331, the first application 234 first decompresses the first message, and then performs the message reassembly operation.

[0079] For another example, in an embodiment, to improve data security, in S301, the message delivered by the application server 210 to the push server 220 is encrypted data (end-to-end encrypted information). Alternatively, in S301, the message delivered by the application server 210 to the push server 220 is unencrypted data. After receiving the message, the push server 220 encrypts the message (end-to-end encrypted information), and then pushes the encrypted message to the terminal device 230.

[0080] In S331, the first application 234 first invokes an encryption/decryption interface to decrypt the first message, and then performs the message reassembly operation.

[0081] In this way, the push agent 232 does not participate in a decryption operation, and therefore cannot obtain specific content of the first message. In an entire process from receiving the pushed first message to displaying the second message, the push agent 232 can obtain only content obtained after message reassembly, thereby greatly improving message security.

[0082] Further, to avoid message leakage, in an embodiment, in S331, if an exception occurs when the first application 234 performs the message reassembly operation (for example, an exception occurs when the message reassembly code is executed), the following is performed after S331: preventing display of the second message (not displaying the notification message), and reporting the failure case to a server, so that an application developer can troubleshoot the problem.

[0083] Further, to enable the first application 234 to implement the message reassembly operation of the second application 236 based on the message reassembly configuration information, in an embodiment, the second application 236 invokes the first application 234 to generate the message reassembly configuration information. For example, the second application 236 invokes the first application 234 to configure message reassembly code and an initialization parameter for the first application 234.

[0084] FIG. 4 is a flowchart of a message processing method according to an embodiment of this application.

[0085] Before S310, the terminal device 230 performs the following procedure shown in FIG. 4 to obtain the message reassembly code and the initialization parameter for the first application 234.

[0086] S400. The second application 236 applies to the push agent 232 for a push token (PushToken).

[0087] S410. After receiving the application of the second application 236, the push agent 232 sets to allow the second application 236 to invoke the first application 234. Specifically, the second application 236 is authorized to be also used as a client applicant (Client Applicant, CA) that can invoke a trusted application (first application 234).

[0088] In addition, the push agent 232 returns the push token to the second application 236.

[0089] Further, to improve security, in an embodiment, in S410, the push agent 232 obtains, from a cloud, a range of resources and interfaces (API) that the first application 234 is allowed to access. When running the message reassembly code, the first application 234 is limited by the range of accessed resources and interfaces. For example, the first application 234 is limited from accessing a network to download a resource such as an icon of the first message, and may only encrypt/decrypt the first message or perform localized formatting of message content text for the first message.

[0090] In S331, the first application 234 performs the message reassembly operation on the first message based on the message reassembly configuration information under a limitation of the range of resources and interfaces that the first application 234 is allowed to access, to generate the second message.

[0091] Based on the limitation of the range of accessed resources and interfaces, in a process in which the message reassembly operation is performed in the first application 234 (TA), only a controllable interface such as a basic data operation, encryption/decryption (API), and a corresponding application resource query is provided, and the first application 234 cannot pull up another application and access another controlled resource, thereby improving system running security.

[0092] S420. After the second application 236 obtains the push token, the second application 236 invokes the first application 234, to set the message reassembly configuration information (the message reassembly code and the initialization parameter) for the first application 234.

[0093] S430. The first application 234 stores the message reassembly configuration information (the message reassembly code and the initialization parameter) to the trusted credential 235 of the first application 234.

[0094] Based on the method in the embodiments of this application, an embodiment of this application further provides a message processing apparatus, and the apparatus is applied to a terminal device.

[0095] FIG. 5 is a schematic diagram of a message processing apparatus according to an embodiment of this application. As shown in FIG. 5, a message processing apparatus 500 includes:

a receiving module 510, configured to receive a first message, where specifically, in an embodiment, with reference to the push agent 232, the receiving module 510 is constructed in the push agent 232;
a message reassembly module 520, configured to pull up a first application module when the first message is a reassembly message, where specifically, in an embodiment, with reference to the push agent 232, the message reassembly module 520 is constructed in the push agent 232; and
the first application module 530 (with reference to the first application 234), configured to:

read message reassembly configuration information (in an embodiment, the message reassembly configuration information is stored in the trusted credential 235), where the message reassembly configuration information is information configured by a second application module (with reference to the second application 236) for the first application module 530, an operation capable of being implemented by the second application module includes at least a message reassembly operation, and the message reassembly configuration information is used to support the first application module 530 in implementing the message reassembly operation; and
perform the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message.

[0096] In the description of this embodiment of this application, for ease of description, when the apparatus is described, the apparatus is divided into various modules based on functions for description, and the division of the modules is only logical function division. When this embodiment of this application is implemented, the functions of the modules may be implemented in one or more pieces of software and/or hardware.

[0097] Specifically, in actual implementation, the apparatus provided in this embodiment of this application may be all or partially integrated into one physical entity, or may be physically separated. In addition, all these modules may be implemented by software invoked by a processing element, or may be implemented by hardware; or some modules may be implemented by software invoked by a processing element, and some modules are implemented by hardware. For example, a detection module may be a separate processing element, or may be integrated into a chip of the electronic device for implementation. The implementation of another module is similar thereto. In addition, all or some of these modules may be integrated together, or may be implemented independently. In an implementation process, the steps of the foregoing method or the foregoing modules may be implemented by using an integrated logic circuit of hardware in the processor element or an instruction in a form of software.

[0098] For example, these foregoing modules may be one or more integrated circuits configured to implement the foregoing method, such as one or more application-specific integrated circuits (Application Specific Integrated Circuit, ASIC), one or more digital signal processors (Digital Signal Processor, DSP), or one or more field programmable gate arrays (Field Programmable Gate Array, FPGA). For another example, these modules may be integrated together, and implemented in a form of a system-on-a-chip (System-On-a-Chip, SOC).

[0099] An embodiment of this application further provides an electronic device (a terminal device). The electronic device includes a memory configured to store a computer program instruction and a processor configured to execute a program instruction. When the computer program instruction is executed by the processor, the electronic device is triggered to perform the method procedure described in the embodiments of this application.

[0100] Specifically, in an embodiment of this application, the foregoing one or more computer programs are stored in the foregoing memory, the foregoing one or

more computer programs include instructions, and when the foregoing instructions are executed by the foregoing device, the foregoing device is enabled to perform the steps of the method described in the embodiments of this application.

[0101] FIG. 6 is a schematic diagram of an electronic device according to an embodiment of this application. An electronic device 600 includes a memory 610 configured to store a computer program instruction and a processor 620 configured to execute a program instruction. When the computer program instruction is executed by the processor 620, the processor 620 controls the electronic device 600 to perform the stylus state switching method described in the embodiments of this application.

[0102] Specifically, in an embodiment of this application, the processor of the electronic device may be a system-on-a-chip SOC, and the processor may include a central processing unit (Central Processing Unit, CPU), and may further include another type of processor. Specifically, in an embodiment of this application, the processor of the electronic device may be a PWM control chip.

[0103] Specifically, in an embodiment of this application, the involved processor may include, for example, a CPU, a DSP, a microcontroller, or a digital signal processor, and may further include a GPU, an embedded neural-network processing unit (Neural-network Process Units, NPU), and an image signal processor (Image Signal Processing, ISP). The processor may further include a necessary hardware accelerator or logic processing hardware circuit, such as an ASIC, one or more integrated circuits configured to control program execution of a technical solutions in this application, or the like. In addition, the processor may have a function of operating one or more software programs, and the software programs may be stored in a storage medium.

[0104] Specifically, in an embodiment of this application, the memory of the electronic device may be a read-only memory (read-only memory, ROM), another type of static storage device that can store static information and instructions, a random access memory (random access memory, RAM), or another type of dynamic storage device that can store information and instructions; or may be an electrically erasable programmable read-only memory (electrically erasable programmable read-only memory, EEPROM), a compact disc read-only memory (compact disc read-only memory, CD-ROM) or another compact disc storage, an optical disc storage (including a compact disc, a laser disc, an optical disc, a digital versatile disc, a Blu-ray disc, or the like), a magnetic disk storage medium, or another magnetic storage device; or may be any computer-readable medium that can be used to carry or store expected program code in a form of an instruction or a data structure and that can be accessed by a computer.

[0105] Specifically, in an embodiment of this application, the processor and the memory may be combined into a processing apparatus, and more commonly, may be components independent of each other. The processor is configured to execute program code stored in the memory to implement the method described in the embodiments of this application. During specific implementation, the memory may be integrated in the processor, or may be independent of the processor.

[0106] Further, the device, apparatus, and modules illustrated in the embodiments of this application may be specifically implemented by a computer chip or entity, or may be implemented by a product with a specific function.

[0107] A person skilled in the art should understand that the embodiments of this application may be provided as a method, an apparatus, or a computer program product. Therefore, the present invention can use a form of hardware only embodiments, software only embodiments, or embodiments with a combination of software and hardware. In addition, the present invention can use a form of a computer program product that is implemented on one or more computer-usable storage media that include computer-usable program code.

[0108] In several embodiments provided in this application, if any function is implemented in a form of a software functional unit and sold or used as an independent product, the function may be stored in a computer-readable storage medium. Based on such an understanding, the technical solutions of this application essentially, or the part contributing to the conventional technology, or some of the technical solutions may be embodied in a form of a software product. The computer software product is stored in a storage medium, and includes several instructions for instructing a computer device (which may be a personal computer, a server, a network device, or the like) to perform all or some of the steps of the methods described in the embodiments of this application.

[0109] Specifically, an embodiment of this application further provides a computer-readable storage medium, where the computer-readable storage medium stores a computer program, and when the computer program runs on a computer, the computer is enabled to perform the method provided in the embodiments of this application.

[0110] An embodiment of this application further provides a computer program product, where the computer program product includes a computer program, and when the computer program is run on a computer, the computer is enabled to perform the method provided in the embodiments of this application.

[0111] The embodiments of this application are described with reference to the flowcharts and/or block diagrams of the method, the device (apparatus), and the computer program product according to embodiments of this application. It should be understood that computer program instructions may be used to implement each procedure and/or each block in the flowcharts and/or the block diagrams and a combination of a procedure and/or a block in the flowcharts and/or the block diagrams. These computer program instructions may be provided to a general-purpose computer, a special-purpose computer, an embedded processor, or a processor of another programmable data processing device to generate a ma-

chine, so that the instructions executed by the computer or the processor of the another programmable data processing device generate an apparatus for implementing a specific function in one or more procedures in the flowcharts and/or in one or more blocks in the block diagrams.

[0112] These computer program instructions may also be stored in a computer-readable memory that can instruct the computer or the another programmable data processing device to work in a specific manner, so that the instructions stored in the computer-readable memory generate an artifact that includes an instruction apparatus. The instruction apparatus implements a specific function in one or more procedures in the flowcharts and/or in one or more blocks in the block diagrams.

[0113] These computer program instructions may also be loaded onto the computer or the another programmable data processing device, so that a series of operations and steps are performed on the computer or the another programmable device, thereby generating computer-implemented processing. Therefore, the instructions executed on the computer or the another programmable device provide steps for implementing a specific function in one or more procedures in the flowcharts and/or in one or more blocks in the block diagrams.

[0114] It should also be noted that, in the embodiments of this application, "at least one" means one or more, and "a plurality of" means two or more. The term "and/or" describes an association relationship of associated objects, and indicates that three relationships may exist. For example, A and/or B may indicate the following three cases: Only A exists, both A and B exist, and only B exists. A and B may be singular or plural. The character "/" usually indicates an "or" relationship between associated objects. "At least one of the following" or a similar expression thereof is any combination of these items, including a single item or any combination of a plurality of items. For example, at least one of a, b, and c may represent a, b, c, a and b, a and c, b and c, or a, b, and c, where a, b, and c may be singular or plural.

[0115] In the embodiments of this application, the term "include", "comprise", or any other variant thereof is intended to cover non-exclusive inclusion, so that a process, method, commodity, or device that includes a series of elements includes not only those elements but also other elements that are not expressly listed, or includes elements inherent to such a process, method, commodity, or device. Without further limitation, the element defined by the sentence "including a" does not exclude that other identical elements also exist in the process, method, commodity, or device including the element.

[0116] This application may be described in the general context of computer-executable instructions, such as a program module, executed by a computer. Generally, the program module includes a routine, a program, an object, a component, a data structure, and the like that perform a specific task or implement a specific abstract data type. Alternatively, this application may be practiced in distributed computing environments. In these distributed computing environments, a task is performed by a remote processing device connected by using a communication network. In the distributed computing environments, the program module may be located in local and remote computer storage media including storage devices.

[0117] Embodiments of this application are described in a progressive manner, and for the same and similar parts of embodiments, reference may be made to each other. Each embodiment focuses on a difference from other embodiments. Especially, for the apparatus embodiment, the apparatus embodiment is basically similar to the method embodiment, and therefore is described briefly. For related parts, refer to the partial description of the method embodiment.

[0118] A person of ordinary skill in the art may be aware that the units and algorithm steps described in the embodiments of this application can be implemented by electronic hardware or a combination of computer software and electronic hardware. Whether these functions are performed by hardware or software depends on specific applications and design constraints of the technical solutions. A person skilled in the art may use different methods to implement the described functions for each specific application, but it should not be considered that such an implementation goes beyond the scope of this application.

[0119] It may be clearly understood by a person skilled in the art that, for convenience and brevity of description, for a specific working process of the described apparatus, apparatus, and unit, refer to a corresponding process in the foregoing method embodiments. Details are not described herein again.

[0120] The foregoing descriptions are merely specific implementations of this application, and any change or replacement readily figured out by a person skilled in the art within the technical scope disclosed in this application shall fall within the protection scope of this application. The protection scope of this application shall be subject to the protection scope of the claims.

Claims

1. A message processing method, wherein the method is applied to a terminal device, and the method comprises:

   receiving a first message; and
   pulling up a first application when the first message is a reassembly message, and the first application performs message reassembly on the first message, comprising:

   reading, by the first application, message reassembly configuration information, wherein the message reassembly configu-

ration information is information configured by a second application for the first application, an operation capable of being implemented by the second application comprises at least a message reassembly operation, and the message reassembly configuration information is used to support the first application in implementing the message reassembly operation; and
performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message.

2. The method according to claim 1, wherein the first application is a trusted application.

3. The method according to claim 2, wherein the message reassembly configuration information is stored in a trusted credential of the first application.

4. The method according to claim 1, wherein the message reassembly configuration information comprises message reassembly code and an initialization parameter; and
the performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message comprises:
executing, by the first application, the message reassembly code for the first message based on the initialization parameter, to generate the second message.

5. The method according to claim 1, wherein before the receiving a first message, the method further comprises:
invoking, by the second application, the first application, to set the message reassembly configuration information.

6. The method according to claim 1, wherein before the receiving a first message, the method further comprises: obtaining, from a cloud, a range of resources and interfaces that the first application is allowed to access; and
the performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message comprises:
performing, by the first application, the message reassembly operation on the first message based on the message reassembly configuration information under a limitation of the range of resources and interfaces that the first application is allowed to access, to generate the second message.

7. The method according to claim 6, wherein the obtaining, from a cloud, a range of resources and interfaces that the first application is allowed to access comprises: obtaining, by a push agent from the cloud, the range of resources and interfaces that the first application is allowed to access.

8. The method according to any one of claims 1-6, wherein the method further comprises:
outputting the second message for display.

9. The method according to claim 8, wherein the outputting the second message for display comprises:

outputting, by the first application, the second message to a push agent; and
outputting, by the push agent, the second message to a notification manager for message display.

10. The method according to claim 8, wherein the method further comprises:
if an exception occurs when the first application performs the message reassembly operation, preventing display of the second message.

11. The method according to claim 1, wherein

the receiving a first message comprises: receiving, by a push agent, the first message; and
the invoking, when the first message is a reassembly message, a first application to perform message reassembly on the first message comprises: when the push agent determines that the first message is a reassembly message, pulling up, by the push agent, the first application, so that the first application performs message reassembly on the first message.

12. The method according to claim 11, wherein before the receiving a first message, the method further comprises:

applying, by the second application, to the push agent for a push token;
after receiving the application of the second application, setting, by the push agent, to allow the second application to invoke the first application, and returning, by the push agent, the push token to the second application;
after the second application obtains the push token, invoking, by the second application, the first application, to set the message reassembly configuration information for the first application; and
storing, by the first application, the message reassembly configuration information.

13. The method according to claim 1, wherein the first message is end-to-end encrypted information; and in a process in which the first application performs the message reassembly operation on the first message, the first application invokes an encryption/decryption interface to decrypt the first message.

14. A message processing apparatus, wherein the apparatus is applied to a terminal device, and the apparatus comprises:

a receiving module, configured to receive a first message;
a message reassembly module, configured to pull up a first application module when the first message is a reassembly message; and
the first application module, configured to:

read message reassembly configuration information, wherein the message reassembly configuration information is information configured by a second application module for the first application module, an operation capable of being implemented by the second application module comprises at least a message reassembly operation, and the message reassembly configuration information is used to support the first application module in implementing the message reassembly operation; and
perform the message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message.

15. An electronic device, wherein the electronic device comprises a memory configured to store a computer program instruction and a processor configured to execute a computer program instruction, and when the computer program instruction is executed by the processor, the electronic device is triggered to perform the steps of the method according to any one of claims 1-13.

16. A computer-readable storage medium, wherein the computer-readable storage medium stores a computer program, and when the computer program is run on a computer, the computer is enabled to perform the method according to any one of claims 1-13.

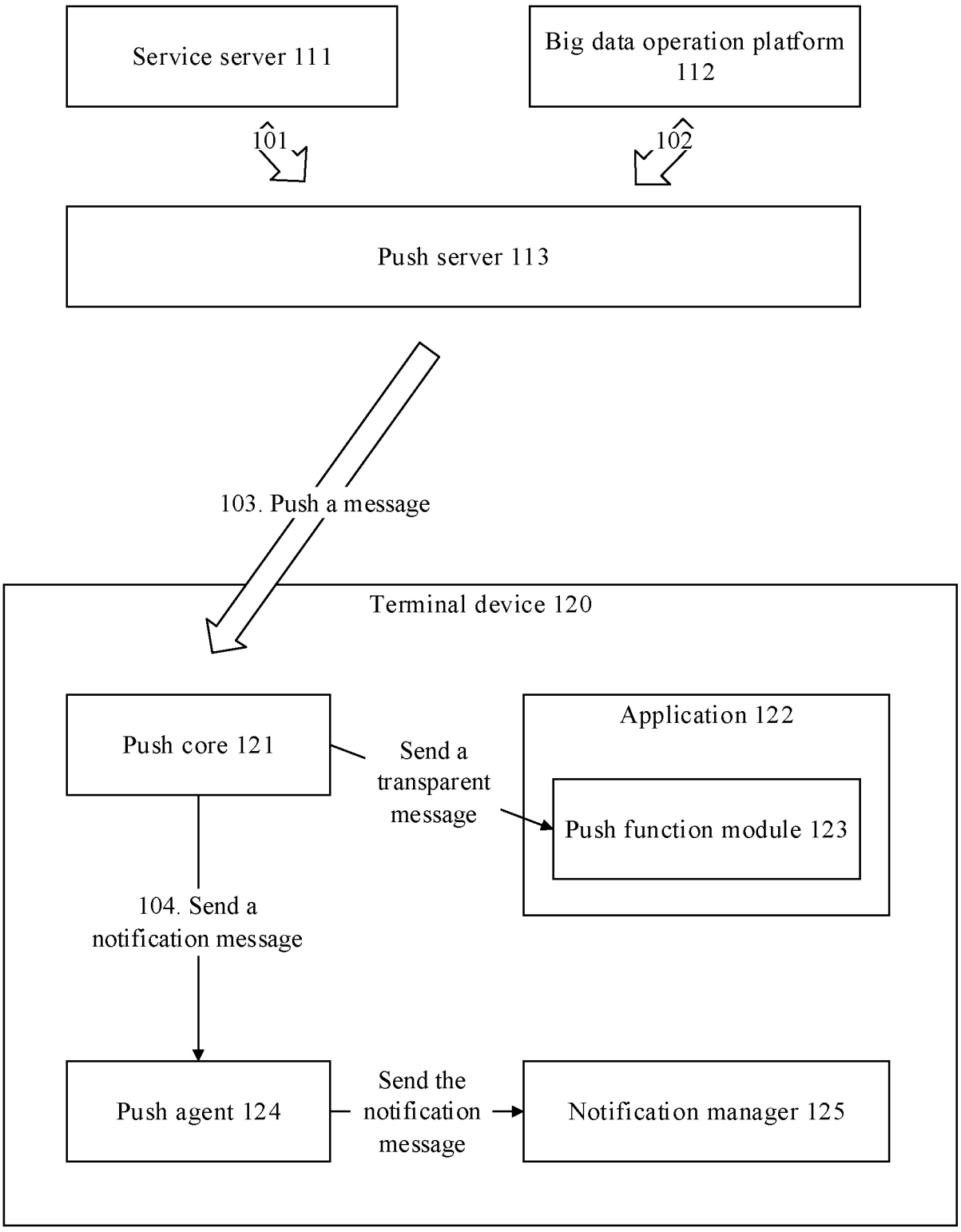
Service server 111
Big data operation platform 112
101
102
Push server 113
103. Push a message
Terminal device 120
Push core 121
Send a transparent message
Application 122
Push function module 123
104. Send a notification message
Push agent 124
Send the notification message
Notification manager 125

FIG. 1

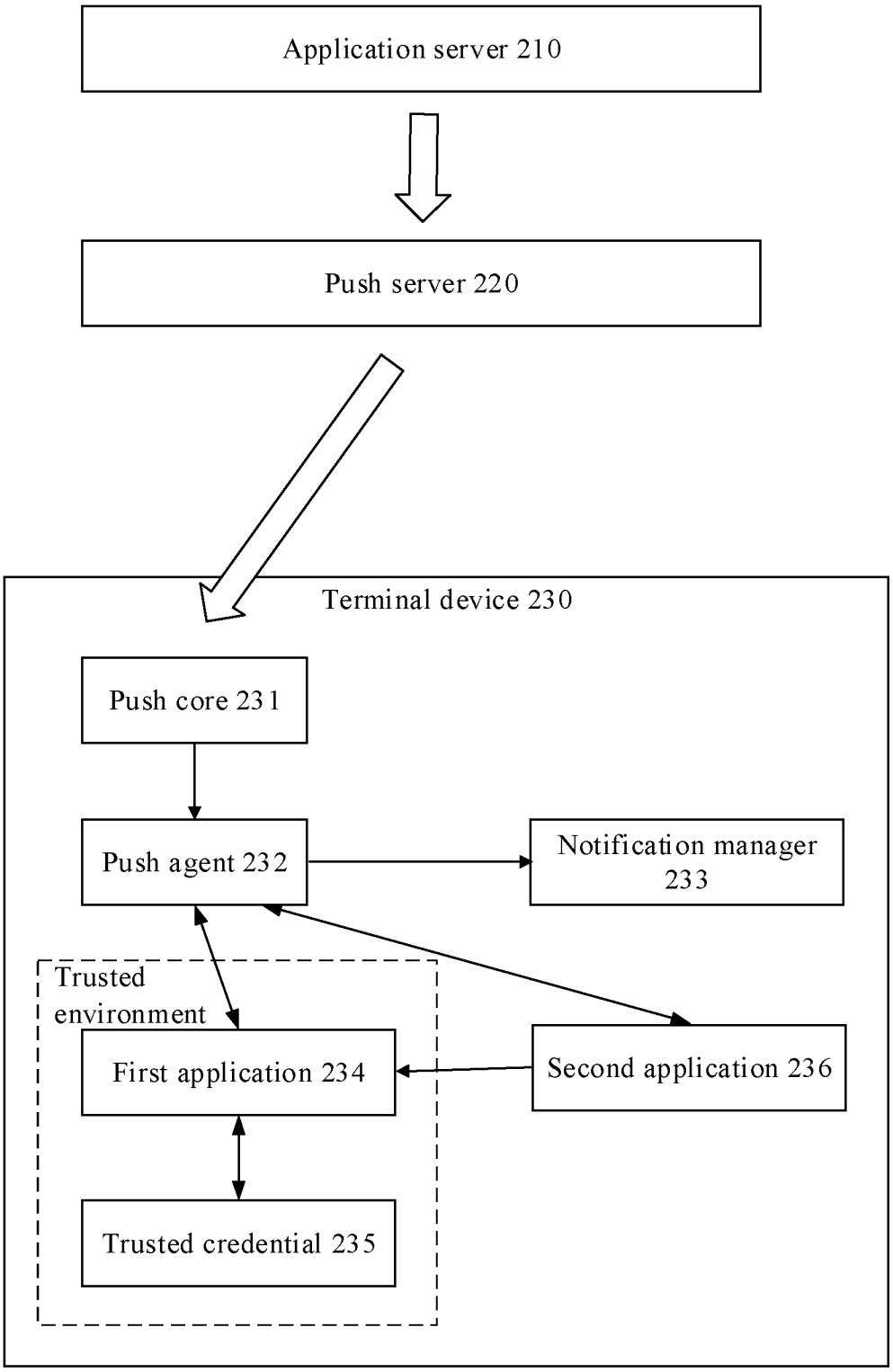
Application server 210
Push server 220
Terminal device 230
Push core 231
Push agent 232
Notification manager 233
Trusted environment
First application 234
Second application 236
Trusted credential 235

FIG. 2

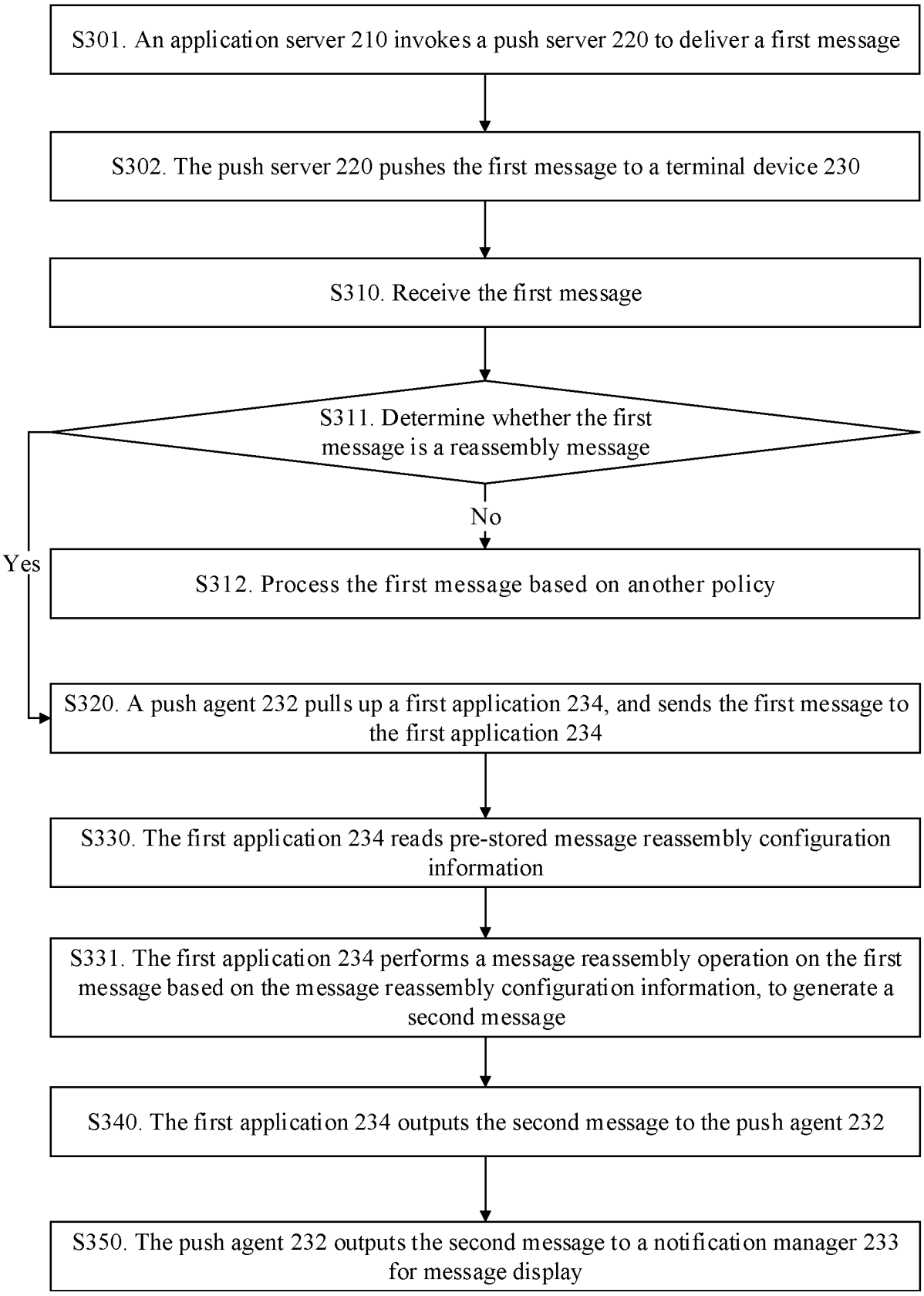
S301. An application server 210 invokes a push server 220 to deliver a first message
S302. The push server 220 pushes the first message to a terminal device 230
S310. Receive the first message
S311. Determine whether the first message is a reassembly message
No
Yes
S312. Process the first message based on another policy
S320. A push agent 232 pulls up a first application 234, and sends the first message to the first application 234
S330. The first application 234 reads pre-stored message reassembly configuration information
S331. The first application 234 performs a message reassembly operation on the first message based on the message reassembly configuration information, to generate a second message
S340. The first application 234 outputs the second message to the push agent 232
S350. The push agent 232 outputs the second message to a notification manager 233 for message display

FIG. 3

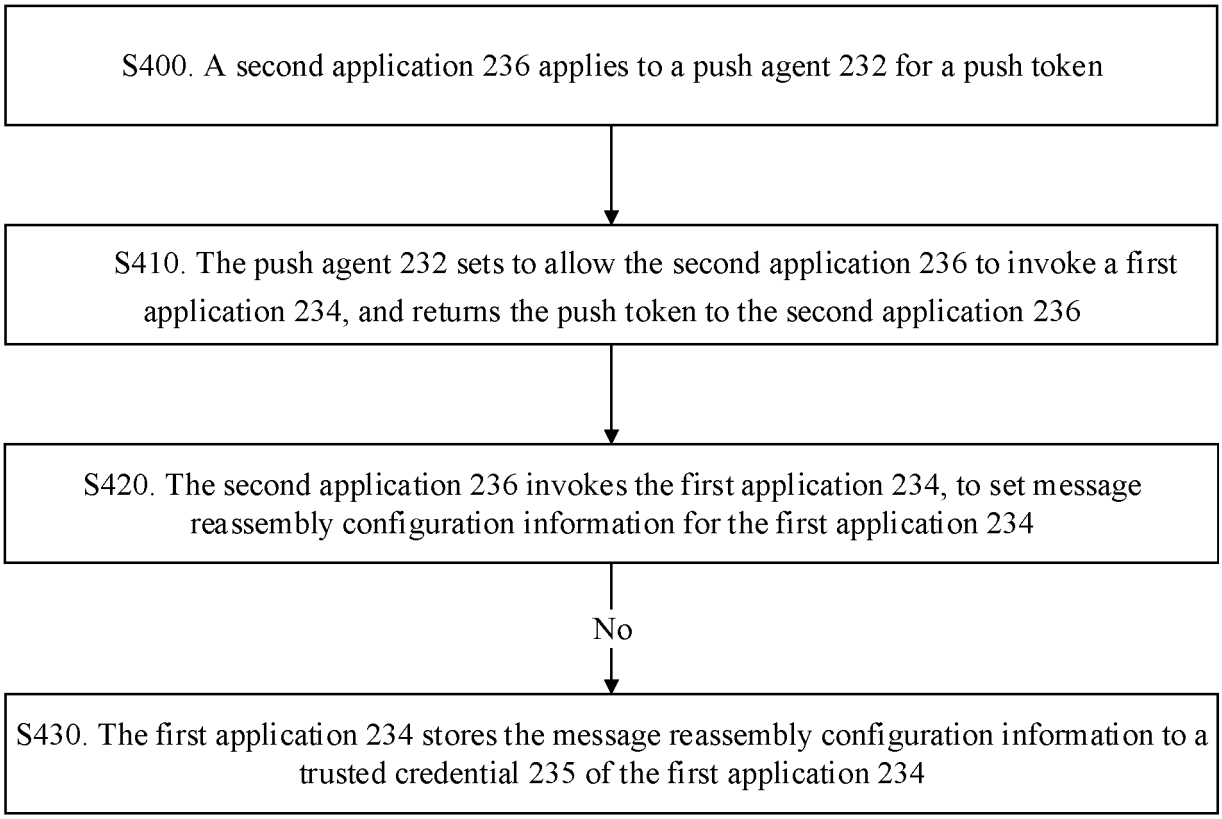
S400. A second application 236 applies to a push agent 232 for a push token
S410. The push agent 232 sets to allow the second application 236 to invoke a first application 234, and returns the push token to the second application 236
S420. The second application 236 invokes the first application 234, to set message reassembly configuration information for the first application 234
No
S430. The first application 234 stores the message reassembly configuration information to a trusted credential 235 of the first application 234

FIG. 4

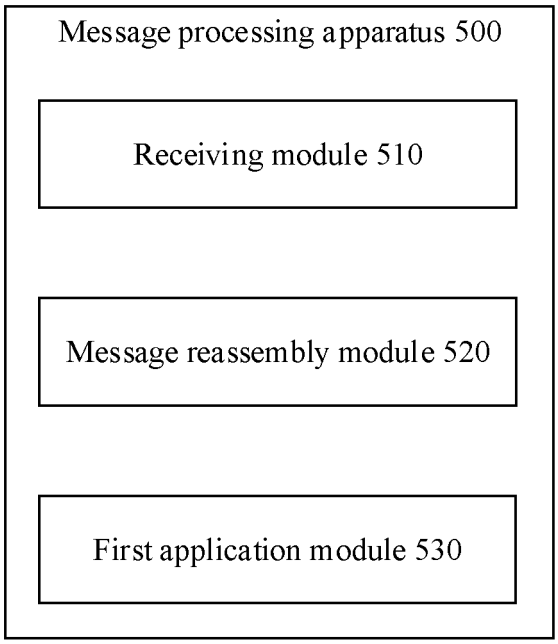
Message processing apparatus 500
Receiving module 510
Message reassembly module 520
First application module 530

FIG. 5

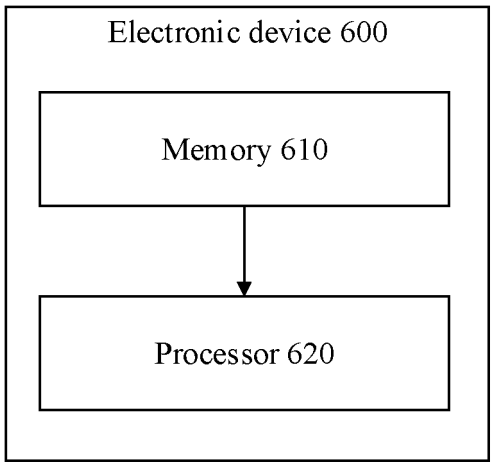

FIG. 6

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2023/088670

A. CLASSIFICATION OF SUBJECT MATTER

G06F21/60(2013.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC:G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

VEN, CNABS, CNTXT, WOTXT, EPTXT, USTXT, CNKI, IEEE, 百度, BAIDU: 华为, 荣耀, 李平, 透传, 通知, ((消息 or 通知) s (加密 or 解密)) and (消息 s 推送) and (明文 or 敏感 or 隐私), 通知栏, 应用, 重组, HUAWEI, HONOR, LI PING, push, message, unvarnished transmission, cleartext, encrypt, decrypt, private, notification bar, application

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CN 114584613 A (HUAWEI TECHNOLOGIES CO., LTD.) 03 June 2022 (2022-06-03) description, paragraphs [0074]-[0080]	1-16
A	CN 110392155 A (ALIBABA GROUP HOLDING LTD.) 29 October 2019 (2019-10-29) entire document	1-16
A	CN 113360238 A (VIVO COMMUNICATION TECHNOLOGY CO., LTD.) 07 September 2021 (2021-09-07) entire document	1-16
A	US 2017359285 A1 (APPLE INC.) 14 December 2017 (2017-12-14) entire document	1-16
A	WO 2018082465 A1 (BEIJING KINGSOFT INTERNET SECURITY SOFTWARE CO., LTD.) 11 May 2018 (2018-05-11) entire document	1-16

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 June 2023	**20 June 2023**
Name and mailing address of the ISA/CN **China National Intellectual Property Administration (ISA/CN)** **China No. 6, Xitucheng Road, Jimenqiao, Haidian District, Beijing 100088**	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (July 2022)

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2023/088670

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN 114584613 A	03 June 2022	None	
CN 110392155 A	29 October 2019	HK 40015691 A0	04 September 2020
CN 113360238 A	07 September 2021	WO 2023274144 A1	05 January 2023
US 2017359285 A1	14 December 2017	US 10554599 B2	04 February 2020
WO 2018082465 A1	11 May 2018	CN 106569668 A	19 April 2017

Form PCT/ISA/210 (patent family annex) (July 2022)